Variations

### Multi-objective knapsack problem

### Abstract: The knapsack problem (KP) and its multidimensional version (MKP) are basic problems in combinatorial optimization. In this paper we consider their multiobjective extension (MOKP and MOMKP), for which the aim is to obtain or to approximate the set of efficient solutions. In a first step, we classify and describe briefly the existing works, that are essentially based on the use of metaheuristics. In a second step, we propose the adaptation of the twophase Pareto local search (2PPLS) to the resolution of the MOMKP. With this aim, we use a very-large scale neighborhood (VLSN) in the second phase of the method, that is the Pareto local search. We compare our results to state-of-the-art results and we show that we obtain results never reached before by heuristics, for the biobjective instances. Finally we consider the extension to three-objective instances.

<https://arxiv.org/ftp/arxiv/papers/1007/1007.4063.pdf?fbclid=IwAR22ua9kDOdnPkl2Ifoz0vpeV9zqAiJCgJvxD4VwLkQpfn0Mm3Cfd63G-pc>

1. **Multi-dimensional knapsack problem**

Abstract: The multidimensional knapsack problem (MDKP) is a knapsack problem with multiple resource constraints. Both the general and the 0-1 versions of this problem have a wide array of practical applications. The MDKP is known to be strongly NP-hard. In this paper, we propose a new greedy-like heuristic method, which is primarily intended for the general MDKP, but proves itself effective also for the 0-1 MDKP. Our heuristic differs from the existing greedy-like heuristics in two aspects. First, existing heuristics rely on each item’s aggregate consumption of resources to make item selection decisions, whereas our heuristic uses the effective capacity, defined as the maximum number of copies of an item that can be accepted if the entire knapsack were to be used for that item alone, as the criterion to make item selection decisions. Second, other methods increment the value of each decision variable only by one unit, whereas our heuristic adds decision variables to the solution in batches and consequently improves computational efficiency significantly for large-scale problems. We conduct intensive numerical studies on randomly generated test problems with a wide range of parameter settings and the benchmark problems from the related literature. We demonstrate that the new heuristic significantly improves computational efficiency of the existing methods and generates robust and near-optimal solutions. The new heuristic proves especially efficient for high dimensional knapsack problems with small-to-moderate numbers of decision variables, usually considered as “hard” MDKP and no computationally efficient heuristic is available to treat such problems.

[**https://www.math.wsu.edu/math/faculty/lih/MDKP.pdf?fbclid=IwAR2kHl06an2IyvLAvwcOAs8inDgg9\_Kk6J5sauJQBQQLCgbS0\_HOVRo8LHw**](https://www.math.wsu.edu/math/faculty/lih/MDKP.pdf?fbclid=IwAR2kHl06an2IyvLAvwcOAs8inDgg9_Kk6J5sauJQBQQLCgbS0_HOVRo8LHw)

### Multiple knapsack problem

The Multiple Knapsack Problem (MKP) is the problem of assigning a subset of n items to m distinct knapsacks, such that the total profit sum of the selected items is maximized, without exceeding the capacity of each of the knapsacks. The problem has several applications in naval as well as financial management. A new exact algorithm for the MKP is presented, which is specially designed for solving large problem instances. The recursive branch-and-bound algorithm applies surrogate relaxation for deriving upper bounds, while lower bounds are obtained by splitting the surrogate solution into the m knapsacks by solving a series of Subset-sum Problems. A new separable dynamic programming algorithm is presented for the solution of Subset-sum Problems, and we also use this algorithm for tightening the capacity constraints in order to obtain better upper bounds. The developed algorithm is compared to the mtm algorithm by Martello and Toth, showing the benefits of the new approach. A surprising result is that large instances with n=100 000 items may be solved in less than a second, and the algorithm has a stable performance even for instances with coefficients in a moderately large range.

[**https://www.sciencedirect.com/science/article/abs/pii/S0377221798001209?fbclid=IwAR3JqvEb\_o9-\_g9ZAEpANFwl6z52jASNaZFLAV88rrWSdMA82v\_WdcrgxBQ#:~:text=The%20Multiple%20Knapsack%20Problem%20**](https://www.sciencedirect.com/science/article/abs/pii/S0377221798001209?fbclid=IwAR3JqvEb_o9-_g9ZAEpANFwl6z52jASNaZFLAV88rrWSdMA82v_WdcrgxBQ#:~:text=The%20Multiple%20Knapsack%20Problem%20)

### Quadratic knapsack problem

### The quadratic knapsack problem (QKP), first introduced in 19th century,[[1]](https://en.wikipedia.org/wiki/Quadratic_knapsack_problem?fbclid=IwAR2U3UCFuy1J8LJb_LCk6TLXzhqVwTKRJB7xgntaWqyH6U4ryx9911MZZJ8#cite_note-1) is an extension of [knapsack problem](https://en.wikipedia.org/wiki/Knapsack_problem) that allows for quadratic terms in the objective function: Given a set of items, each with a weight, a value, and an extra profit that can be earned if two items are selected, determine the number of items to include in a collection without exceeding capacity of the [knapsack](https://en.wikipedia.org/wiki/Knapsack), so as to maximize the overall profit. Usually, quadratic knapsack problems come with a restriction on the number of copies of each kind of item: either 0, or 1. This special type of QKP forms the [0-1 quadratic knapsack problem](https://en.wikipedia.org/wiki/0-1_quadratic_knapsack_problem), which was first discussed by Gallo et al.[[2]](https://en.wikipedia.org/wiki/Quadratic_knapsack_problem?fbclid=IwAR2U3UCFuy1J8LJb_LCk6TLXzhqVwTKRJB7xgntaWqyH6U4ryx9911MZZJ8#cite_note-2) The 0-1 quadratic knapsack problem is a variation of knapsack problems, combining the features of unbounded knapsack problem, 0-1 knapsack problem and quadratic knapsack problem.

### <https://en.wikipedia.org/wiki/Quadratic_knapsack_problem?fbclid=IwAR2U3UCFuy1J8LJb_LCk6TLXzhqVwTKRJB7xgntaWqyH6U4ryx9911MZZJ8>

### Subset-sum problem

The **subset sum problem** (SSP) is a [decision problem](https://en.wikipedia.org/wiki/Decision_problem) in [computer science](https://en.wikipedia.org/wiki/Computer_science). In its most general formulation, there is a [multiset](https://en.wikipedia.org/wiki/Multiset) {\displaystyle S}of integers and a target-sum {\displaystyle T}, and the question is to decide whether any subset of the integers sum to precisely {\displaystyle T}*.*[[1]](https://en.wikipedia.org/wiki/Subset_sum_problem?fbclid=IwAR0UkiNO6XymsBXbp6YNn_4QSczM3-wPIMvWUN8VpBuVobgeS_VKuraeUDQ#cite_note-kleinberg2006p491-1) The problem is known to be NP. Moreover, some restricted variants of it are [NP-complete](https://en.wikipedia.org/wiki/NP-completeness) too, for example:[[1]](https://en.wikipedia.org/wiki/Subset_sum_problem?fbclid=IwAR0UkiNO6XymsBXbp6YNn_4QSczM3-wPIMvWUN8VpBuVobgeS_VKuraeUDQ#cite_note-kleinberg2006p491-1)

* The variant in which all inputs are positive.
* The variant in which inputs may be positive or negative, and {\displaystyle T=0}. For example, given the set {\displaystyle \{-7,-3,-2,9000,5,8\}}, the answer is *yes* because the subset {\displaystyle \{-3,-2,5\}} sums to zero.
* The variant in which all inputs are positive, and the target sum is exactly half the sum of all inputs, i.e., {\displaystyle T={\frac {1}{2}}(a\_{1}+\dots +a\_{n})} . This special case of SSP is known as the [partition problem](https://en.wikipedia.org/wiki/Partition_problem).

SSP can also be regarded as an [optimization problem](https://en.wikipedia.org/wiki/Optimization_problem): find a subset whose sum is at most *T*, and subject to that, as close as possible to *T*. It is [NP-hard](https://en.wikipedia.org/wiki/NP-hard), but there are several algorithms that can solve it reasonably quickly in practice.

SSP is a special case of the [knapsack problem](https://en.wikipedia.org/wiki/Knapsack_problem) and of the [multiple subset sum](https://en.wikipedia.org/wiki/Multiple_subset_sum) problem.

[**https://en.wikipedia.org/wiki/Subset\_sum\_problem?fbclid=IwAR0UkiNO6XymsBXbp6YNn\_4QSczM3-wPIMvWUN8VpBuVobgeS\_VKuraeUDQ**](https://en.wikipedia.org/wiki/Subset_sum_problem?fbclid=IwAR0UkiNO6XymsBXbp6YNn_4QSczM3-wPIMvWUN8VpBuVobgeS_VKuraeUDQ)

1. **Geometric knapsack problem**

The objective of this paper is to present an experimental study of the Geometric Knapsack Problem (GKP) with the goal of obtaining provably optimal solutions. We introduce an Integer Linear Programming model for the GKP and apply it to hundreds of instances of two classes: one comprised of uniformly generated points with randomly assigned values; and another composed of convex layered points with value distribution biased towards concentrating negative-valued points on the innermost layers. Trial tests were used to guide the choice of input parameters so as to avoid generating trivial instances. Our experiments show that the layered class is significantly harder to be solved to optimality, in practice, since even instances with as few as 35 points could not be solved within 5 minutes of CPU time.

[**http://www.cs.umanitoba.ca/~cccg2018/papers/session5B-p2.pdf**](http://www.cs.umanitoba.ca/~cccg2018/papers/session5B-p2.pdf)